```
Discussion : 2020-07-20
-----------------------
Saurab
------
    1.
    2. gram indices : abcus abc -> 450, bcu -> 679
    vector dimension: 1 5000

    grams [abc, bcu, cus, ab, bc, cu, us]
    to avoid iterating from 1 to 5000.

Shankar
-------
    1. Index Map :
    2. Completed without query search in relevant words.
    3. How to implement with relevant words ?

    vector of entry =>  n = 15
    no. of words =>     1000 = n + k

    (n + k) * n for each wrong word.
    n^2 + nk => O(n^2)

    Inverted index =>
        g1 -> [w1, w203, w500]
        g2 -> [w1, w609, w999]


    During Search:
    wrong_word wk = [g100, g2, g890, g350]

    wk -> map with inverted_index -> List of List
    filter distinct -> list of words (relevant)

--------
What is log function ?
 2 ^ 5 = 32
 log2(32) = 5

----------------

Minu
------
    1. flow completed.
    2.

w1: apple => [ap, pp, pl, le, app, ppl, ple]
w2:
w3:

w1000:

First transformation.
flatten with value,
apple -> ap
apple -> pp
apple -> pl
apple -> le
apple -> app
apple -> ppl
apple -> ple
.
.
.
```

```

Tranformation group by value,
ap -> [(apple -> ap), (appliation -> ap), ...]
pp -> [(), ()]

.
.
.

map apply to values tuple
ap -> [apple, applicaton, . . .], distinct.

This is inverted index.
idf -> N/count values. [Normalization.]

[1, 0, 5, 6, 9, 0, 0, 8]
[0, 0, 0,          0, 100]
[0, 0, 0,   1, 1, 0, 0, 0]

----------------------------------------
n -> input -> f(n) -> ? to compute.

f:hash(n) => K constant time.
formula, -> K, retrieve using k. O(1)
f:max(n) -> linear -> O(n)
sorting -> O(nlogn) quick sort, merge sort, n^2

finding index in list(vector):
["Minu", "Astha", "Shankar", "Ayush", "Saurab"]

query("Astha") -> 1
query("Santa") -> -1
O(n)
-----------------------------------------------
sort
binary search
O(logn)
-----------------------------------------------
HashSet         -> ? unordered. isPresent?  O(1)
Binary search   -> O(logn)
Seq. search     -> O(n)

n log n
-------
Fib Series
----------

Give me nth Fib number ?
-------------------------
N^2
NlongN
-------------------------
normal recursion. n^2

Memomization
--------------
NlogN
-------------- You can check it.

Astha
--------------

[a, b, c, a, b, a]
trainsform to
```

```
[a -> 1, b -> 1, c -> 1, a -> 1, b -> 1, a -> 1]

group_by
first key,

[a [1, 1, 1], b [1, 1], c [1], d [1]]

sum value.
[a 3, b 2, c 1, d 1]

Total data N
3 N
O(N) => linear time.

-----------------------------------------------
zip function
-----------------------------------------------
x = [a, b, a, b, a, a]
y = [1, 1, 1, 1, 1, 1]
-----------------------------------------------
zip function
zip(x,y) => [(a, 1), (b, 1), (a, 1), (b, 1) ...]

-----------------------
[a, b]
[0 a, 1 b] => hashmap
----------------------
Binod
------
    1. trying to learn data transformation with other data.

-----------------------------------------------
```